1.2 Pensamiento computacional: evolución conceptual

La primera referencia formal de pensamiento computacional la podemos encontrar en el artículo de **Wing** (2006), donde define el pensamiento computacional como la **habilidad que implica resolver problemas, diseñar sistemas, y entender el comportamiento humano** a partir de los conceptos fundamentales de la informática, y que incluye una gama de herramientas mentales que reflejan la amplitud del campo de la informática. La autora argumenta no sólo lo que es el pensamiento computacional, sino también lo que no es:

1. *"Conceptualizing, not programming" (conceptualizar, no programar').* Pensar como un programador de dispositivos va mucho más allá de estar capacitado para programar un ordenador, ya que ello requiere tener la capacidad de pensar en múltiples capas de abstracción.
2. *"A way that humans, not computers, think" ('una manera en que los humanos piensan, no las computadoras').* El pensamiento computacional es una habilidad que las personas usamos para resolver problemas, no para simular el pensamiento de un ordenador. Las máquinas están a nuestra disposición y su forma de realizar las cosas es predecible. Sin embargo, las personas, tenemos la capacidad de ser creativos, inteligentes y espontáneos. Por lo tanto, somos los humanos los que creamos estas máquinas para que nos ayuden, utilizando para ello nuestra inteligencia para acometer y resolver problemas que seguramente seríamos incapaces de poder realizar antes de inventar las máquinas.
3. *"Fundamental, not rote skill" (habilidad básica, no puramente mecánica).* Se considera una habilidad básica aquella que cualquier ser humano tiene que poseer para poder desenvolverse en esta sociedad actual.
4. *"Complements and combines mathematical and engineering thinking" (se complementa y se combina con el pensamiento matemático e ingeniero).* El pensamiento computacional tiene una relación en su origen con el pensamiento matemático, como el resto de ciencias. De la misma forma, tiene una relación con el pensamiento desarrollado en estudios de ingeniería puesto que lo que se crean son construcciones de sistemas informáticos para interactuar con nuestro mundo físico.

Copyright 2026

5. *"Ideas, not artifacts" (ideas, no artefactos).* El pensamiento computacional no sólo está relacionado con las creaciones hardware o software que el ser humano sea capaz de diseñar, sino que también es una habilidad que podemos usar siempre para resolver problemas tan cotidianos como preparar un plato en la cocina o para gestionar mejor nuestra agenda personal.
6. *"For everyone, everywhere" (para cualquiera, en cualquier parte).* El pensamiento computacional será una realidad cuando lo tengamos tan integrado en nuestras formas de abordar tareas, que lo más lógico sea que desaparezca como término y filosofía explícitos.

La Doctora Wing actualizará su propia definición argumentando que el pensamiento computacional incluye los procesos de pensamiento implicados en la formulación de problemas y de sus soluciones, de tal modo que éstos estén representados de una manera que pueda ser abordada efectivamente por un agente-procesador de información (Wing, 2008).

A partir de este momento, se suceden en el tiempo diferentes aportaciones, todas ellas enfocadas a enriquecer los currículos educativos. Fruto del trabajo colaborativo de la "*Computer Science Teachers Association"* (CSTA, 2011) y la "*International Society for Technology in Education*" (ISTE) de los Estados Unidos surge su propia aportación: un enfoque para resolver un problema concreto que ayuda a la inclusión de tecnologías digitales con ideas humanas. Todo ello no reemplaza el énfasis en creatividad, razonamiento o pensamiento crítico pero refuerza esas habilidades al tiempo que realza formas de organizar el problema de manera que el ordenador pueda ayudar (CSTA & ISTE, 2011).

En 2012 la Royal Society (Reino Unido) crea su primera definición al respecto, donde se argumenta que el pensamiento computacional es el proceso de reconocimiento de los aspectos computables en el mundo que nos rodea, y de aplicar las herramientas y técnicas de las Ciencias de la Computación para comprender y razonar sobre sistemas y procesos, tanto naturales como artificiales (Royal Society, 2012).

Otra aportación interesante ha sido la que realizaron Grover y Pea (2013), quienes proponen los principales conceptos que ellos piensan que han generado el mayor consenso, y que por lo tanto, deberían estar presentes en cualquier currículo educativo:

1. Abstracción y generalización de patrones (incluyendo modelos y simulaciones)
2. Procesamiento sistemático de la información
3. Sistemas de símbolos y representación
4. Noción algorítmica de control de flujo
5. Descomposición estructurada de problemas
6. Pensamiento iterativo, recursivo y paralelo
7. Lógica condicional
8. Limitadores de eficiencia y rendimiento

Copyright 2026

9. Depuración y detección sistemática de errores

Kafai y Burke (2014) amplían las definiciones anteriores con un concepto innovador, definiéndolo como un tipo de pensamiento basado en procesos ejecutados por una persona o una máquina utilizando métodos y modelos que permiten resolver problemas así como diseñar sistemas que por sí solos no podrían hacerlo.

El equipo de desarrollo de Scratch (Lamb y Johnson, 2011) el software educativo más utilizado en el mundo (Zhang y Nouri, 2019), aportó su visión definiendo el pensamiento computacional como un conjunto de conceptos, prácticas y perspectivas que está fundamentado en el ámbito de la informática. Para ellos, aprender a programar y compartir sus propias creaciones provoca en los estudiantes que se desarrollen como pensadores computacionales, aprendiendo conceptos básicos a la vez que son capaces de desarrollar estrategias de resolución de problemas, diseño y formas de colaboración (ScratchEd Team, 2015). En la misma línea, es visto como una metodología que implementa conceptos básicos de la computación que ayudan a resolver cualquier clase de problemas, forjar estrategias y ejecutar tareas de tal forma que nos permita afrontar los problemas con eficacia y posibilidades de éxito. (Olabe, Basogain y Basogain, 2015).

Para finalizar este apartado de definiciones, hilaremos la primera de las definiciones realizada por Wing (2006, 2008) con la postura de Bers (2017), la cual destaca que aunque la resolución de problemas tiene su importancia dentro de la definición más operacional del pensamiento computacional, le otorga especial relevancia al hecho de que el principal potencial es la posibilidad de expresar y crear ideas mientras programamos, argumentando que la programación, al igual que la escritura, es una forma de expresarse. Así, si con el lenguaje somos capaces de concretar múltiples y variadas representaciones, con los lenguajes de programación somos capaces también de expresarnos y crear productos (del Mar Sánchez-Vera, 2019).

Todos los conceptos y literatura comentadas en este capítulo están extraídos del trabajo de revisión sistemática sobre métodos de evaluación del pensamiento computacional (Ruiz y Bustamente, 2021).

Bibliografía

Bers, M. U. (2017). *Coding as a playground: Programming and computational thinking in the early childhood classroom.* Routledge.

CSTA (2011). K–12 Computer Science Standards (Level 2) [Documento en línea]. Recuperado de http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K12_CSS.pdf

Copyright 2026

CSTA & ISTE (2011). Operational Definition of Computational Thinking for K–12 Education [Documento en línea]. Recuperado

dehttp://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf

del Mar Sánchez-Vera, M. (2019). El pensamiento computacional en contextos educativos: una aproximacion desde la Tecnologia Educativa/Computational Thinking in Educational Environments: An Approach from Educational Technology/El pensament computacional en contextos educatius: una aproximado des de la Tecnologia Educativa. *Research in Education and Learning Innovation Archives (REALIA), (23),* 24-40.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher, 42(1),* 38-43.

Kafai, Y. B., & Burke, Q. (2014). *Connected code: why children need to learn programming.* MIT Press.

Olabe, X. B., Basogain, M. Á. O., & Basogain, J. C. O. (2015). Pensamiento Computacional a través de la Programación: Paradigma de Aprendizaje. *Revista de Educación a Distancia (RED),* (46)

Royal Society (Great Britain). (2012*). Shut down or restart?: The way forward for computing in UK schools.* Royal Society.

Ruiz Reinales, C., & Bustamante, J. C. Pensamiento computacional en educación infantil y primaria: una revisión sistemática.

ScratchEd Team [Portal Web] (2015). Computational Thinking webinars. Recuperado 2 de Junio de 2015, de http://scratched.gse.harvard.edu/content/1488

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725.

Wing, J. M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology, 25*(2), 7–14.

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education, 141,* 103607.

Financiado por el Ministerio de Educación y Formación Profesional y por la Unión Europea - NextGenerationEU

Copyright 2026



Revision #6
Created 2022-10-08 07:21:21 CEST by Cristian Ruiz
Updated 2023-01-17 15:49:44 CET by Equipo CATEDU

Copyright 2026